ID2223: Scalable Machine Learning and Deep Learning Project: Scalable Distributed Deep Reinforcement Learning for Multiple Stock Trading

Due on 2022/01/08

Tianzhang Cai, Yage Hao

Contents

1 Introduction

Deep reinforcement learning (deep RL) is a subfield of machine learning that combines reinforcement learning (RL) and deep learning. It has been used for a diverse set of applications including but not limited to robotics, video games, natural language processing, computer vision, education, transportation, finance and healthcare [1].

In this project, we intend to design an RL-based automated trading solution for multiple stock trading according to paper FinRL: A Deep Reinforcement Learning Library for Automated Stock Trading in Quantitative Finance, presented at NeurIPS 2020: Deep RL Workshop [2]. As we know, time is extremely valuable in financial trading. In order to achieve scalability and efficiency in multiple stock trading, we will implement a distributed RL algorithm according to the paper IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures [3].

2 How to run?

Requirements:

- Python 3
- Packages: FinRL, Yahoo Finance API, pandas, numpy, matplotlib, stockstats, OpenAI gym, stable-baselines, tensorflow, pyfolio, jupyter

Directly run the Jupyter Notebook "main.ipynb".

3 Implementation

3.1 Preprocessing Data

Our data are from Yahoo Finance [4]. Yahoo Finance is a website that provides stock data, financial news, financial reports, etc. All the data provided by Yahoo Finance is free.

We fetched DOW-30 stock data between 2009-01-01 and 2021-10-31. The raw dataset consists 94331 observations and 8 features. A sample is shown in Figure 1.

	date	open	high	low	close	volume	tic	day
0	2009-01-02	3.067143	3.251429	3.041429	2.778781	746015200	AAPL	4
1	2009-01-02	58.590000	59.080002	57.750000	45.615864	6547900	AMGN	4
2	2009-01-02	18.570000	19.520000	18.400000	15.579443	10955700	AXP	4
3	2009-01-02	42.799999	45.560001	42.779999	33.941097	7010200	BA	4
4	2009-01-02	44.910000	46.980000	44.709999	32.475784	7117200	CAT	4

Figure 1: Samples from raw dataset

In addition to the above features, we introduced technical indicators and turbulence index.

- **Technical indicators:** In practical trading, various information needs to be taken into account, for example the historical stock prices, current holding shares, technical indicators, etc. In this project, we demonstrate two trend-following technical indicators: MACD and RSI.
- **Turbulence index:** Risk-aversion reflects whether an investor will choose to preserve the capital. It also influences one's trading strategy when facing different market volatility level. To control the risk in a worst-case scenario, such as financial crisis of 2007–2008, FinRL employs the financial turbulence index that measures extreme asset price fluctuation.

Finally, we received the prepared dataset with 93641 observations and 18 features as shown in Figure 2.

	date	tic	open	high	low	close	volume	day	macd	boll_ub	boll_lb	rsi_30	cci_30	dx_30	close_30_sma	close_60_sma	vix	turbulence
0	2009-01-02	AAPL	3.067143	3.251429	3.041429	2.778781	746015200.0	4.0	0.0	3.003272	2.671566	100.0	66.666667	100.0	2.778781	2.778781	39.189999	0.0
1	2009-01-02	AMGN	58.590000	59.080002	57.750000	45.615856	6547900.0	4.0	0.0	3.003272	2.671566	100.0	66.666667	100.0	45.615856	45.615856	39.189999	0.0
2	2009-01-02	AXP	18.570000	19.520000	18.400000	15.579447	10955700.0	4.0	0.0	3.003272	2.671566	100.0	66.666667	100.0	15.579447	15.579447	39.189999	0.0
3	2009-01-02	BA	42.799999	45.560001	42.779999	33.941097	7010200.0	4.0	0.0	3.003272	2.671566	100.0	66.666667	100.0	33.941097	33.941097	39.189999	0.0
4	2009-01-02	CAT	44.910000	46.980000	44.709999	32.475807	7117200.0	4.0	0.0	3.003272	2.671566	100.0	66.666667	100.0	32.475807	32.475807	39.189999	0.0
5	2009-01-02	CRM	8.025000	8.550000	7.912500	8.505000	4069200.0	4.0	0.0	3.003272	2.671566	100.0	66.666667	100.0	8.505000	8.505000	39.189999	0.0

Figure 2: Samples from prepared dataset

3.2 Environment Design

First, we need to define our stock trading problem in a reinforcement learning environment. According to FinRL paper, the stock trading process can be modeled as a Markov Decision Process (MDP) while treating the trading goal as a reward maximization problem.

- **Action:** We use an action space −k, ..., −1, 0, 1, ..., k, where k denotes the number of shares, minus sign denotes selling, plus sign denotes buying, 0 denotes holding.
- **Reward function:** Our reward function is designed as $r(s, a, s) = vv$. It means from state s, by taking action a to state s', the reward is the differences of portfolio values.
- **State:** The state space describes the observations that the agent receives from the environment.
- **Environment:** Dow 30.

The training process involves observing stock price change, taking an action and reward's calculation to have the agent adjusting its strategy accordingly. By interacting with the environment, the trading agent will derive a trading strategy with the maximized rewards as time proceeds.

Our trading environments, based on OpenAI Gym framework, simulate live stock markets with real market data according to the principle of time-driven simulation.

3.3 Scalable Distributed Deep Reinforcement Learning Algorithms

The implementation of the distributed agent is based on paper: IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures [3]. The paper demonstrated two state-of-art methods: IMPALA and V-trace.

IMPALA is capable of scaling to thousands of machines without sacrificing training stability or data efficiency. IMPALA actors communicate trajectories of experience (sequences of states, actions, and rewards) to a centralised learner.IMPALA (Figure 3) uses an actor-critic setup to learn a policy π and a baseline function V^{π}. The process of generating experiences is decoupled from learning the parameters of π and V^{π}.

The architecture consists of a set of actors, repeatedly generating trajectories of experience, and one or more learners that use the experiences sent from actors to learn π off-policy.

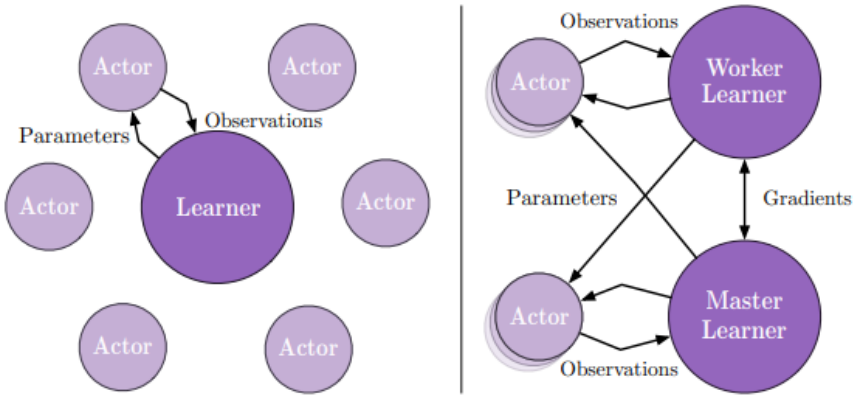

Figure 3: IMPALA Architecture.

However, because the policy used to generate a trajectory can lag behind the policy on the learner by several updates at the time of gradient calculation, learning becomes off-policy. The V-trace off-policy actor-critic algorithm is introduced to correct for this harmful discrepancy.

4 Results

To test how good the model is, assume that we have $1,000,000 initial capital on 2020-07-01. We use the model to trade Dow-30 stocks. We evaluate the performance of our trading strategy by applying automated backtesting tool. The statistical results and backtesting plots are shown below.

Start date	**2020-07-01**
End date	**2021-10-28**
Total months	**16**
	Backtest
Annual return	30.355%
Cumulative returns	42.398%
Annual volatility	14.05%
Sharpe ratio	1.96
Calmar ratio	3.58
Stability	0.95
Max drawdown	-8.49%
Omega ratio	1.39
Sortino ratio	2.99
Skew	NaN
Kurtosis	NaN
Tail ratio	1.02
Daily value at risk	-1.661%
Alpha	0.05
Beta	0.91

Figure 4: Backtest stats.

Worst drawdown periods	Net drawdown in %	Peak date	Valley date	Recovery date	Duration
0	8.49	2020-09-02	2020-09-23	2020-11-09	49
1	5.86	2021-08-23	2021-10-04	2021-10-19	42
2	4.02	2021-02-24	2021-03-04	2021-03-10	11
3	3.48	2021-06-04	2021-06-18	2021-06-25	16
4	3.44	2021-01-08	2021-01-29	2021-02-05	21

Figure 5: Worst drawdown periods stats.

We can see that the stock trading predictions would achieve an average annual return of 30.35%. The total cumulative return is 42.40%.

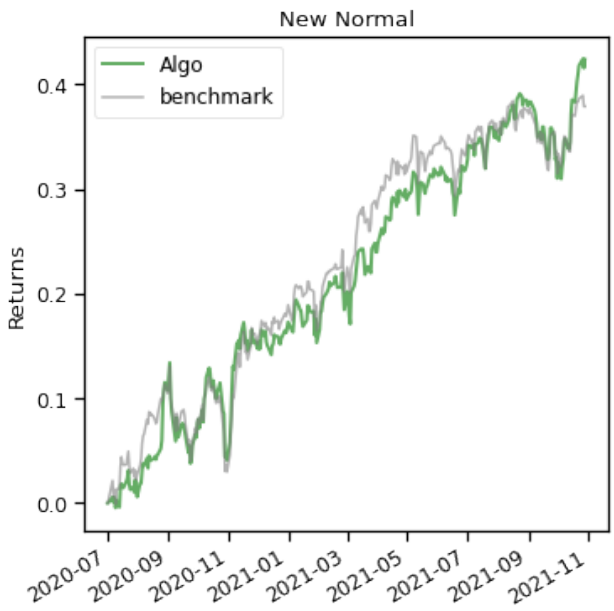

Figure 6: Backtest plot of cumulative returns.

We also compared the algorithm with a baseline stock history retrieved from Yahoo Finance API. We see that the algorithm achieved about 5% more profit compared to the baseline returns.

Finally we plotted more detailed statistics in the backtesting in Figure 7, to help the user to do further analysis with the assistance of the RL algorithm.

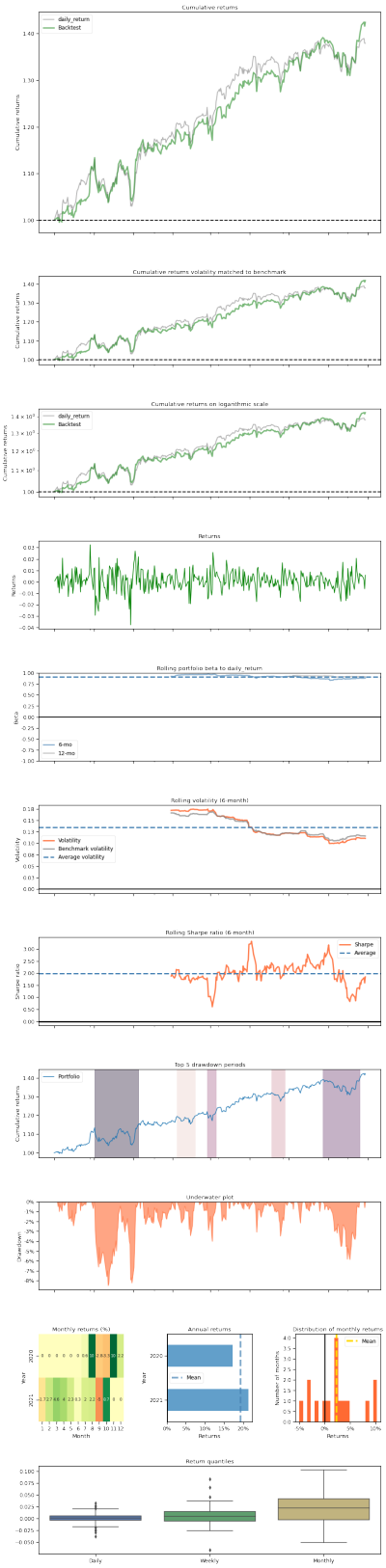

Figure 7: Backtest plots of various statistics.

References

[1] Wikipedia, *Deep Reinforcement Learning*, Available:
https://en.wikipedia.org/wiki/Deep_reinforcement_learning

[2] Xiao-Yang Liu, Hongyang Yang, Qian Chen, Runjia Zhang, Liuqing Yang, Bowen Xiao, Christina Dan Wang, *FinRL: A Deep Reinforcement Learning Library for Automated Stock Trading in Quantitative Finance*, Available:
https://arxiv.org/abs/2011.09607

[3] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Volodymyr Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, Shane Legg, Koray Kavukcuoglu, *IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures*, Available:
https://arxiv.org/pdf/1802.01561.pdf

[4] Yahoo Finance, *Yahoo Finance API Specification*, Available:
https://www.yahoofinanceapi.com/